# 如何编写更省电的Android应用

电池医生经验分享

# 目录

- 为什么要省电？
- 有哪些地方在耗电？
- 怎样编写更省电？
- 性能分析工具介绍

# 为什么要省电？

# 硬件

1. 更加清晰的屏幕

Samsung I9220：5.3英寸 1280x800像素

HTC G23 One X： 4.7英寸 1280x720像素

2. 更加多核的CPU

小米M2： 1.5GHz 四核

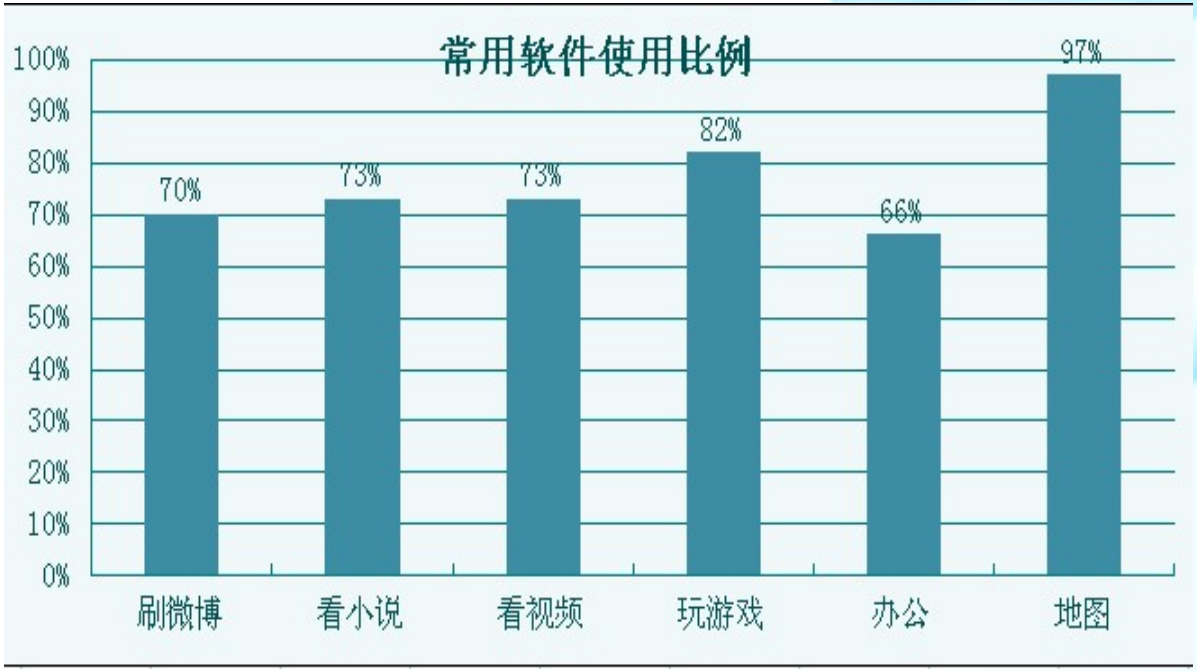
HTC Droid DNA： 1.5GHz 四核

3.千万级像素摄像头

Moto XT928： 1300万像素

# 软件

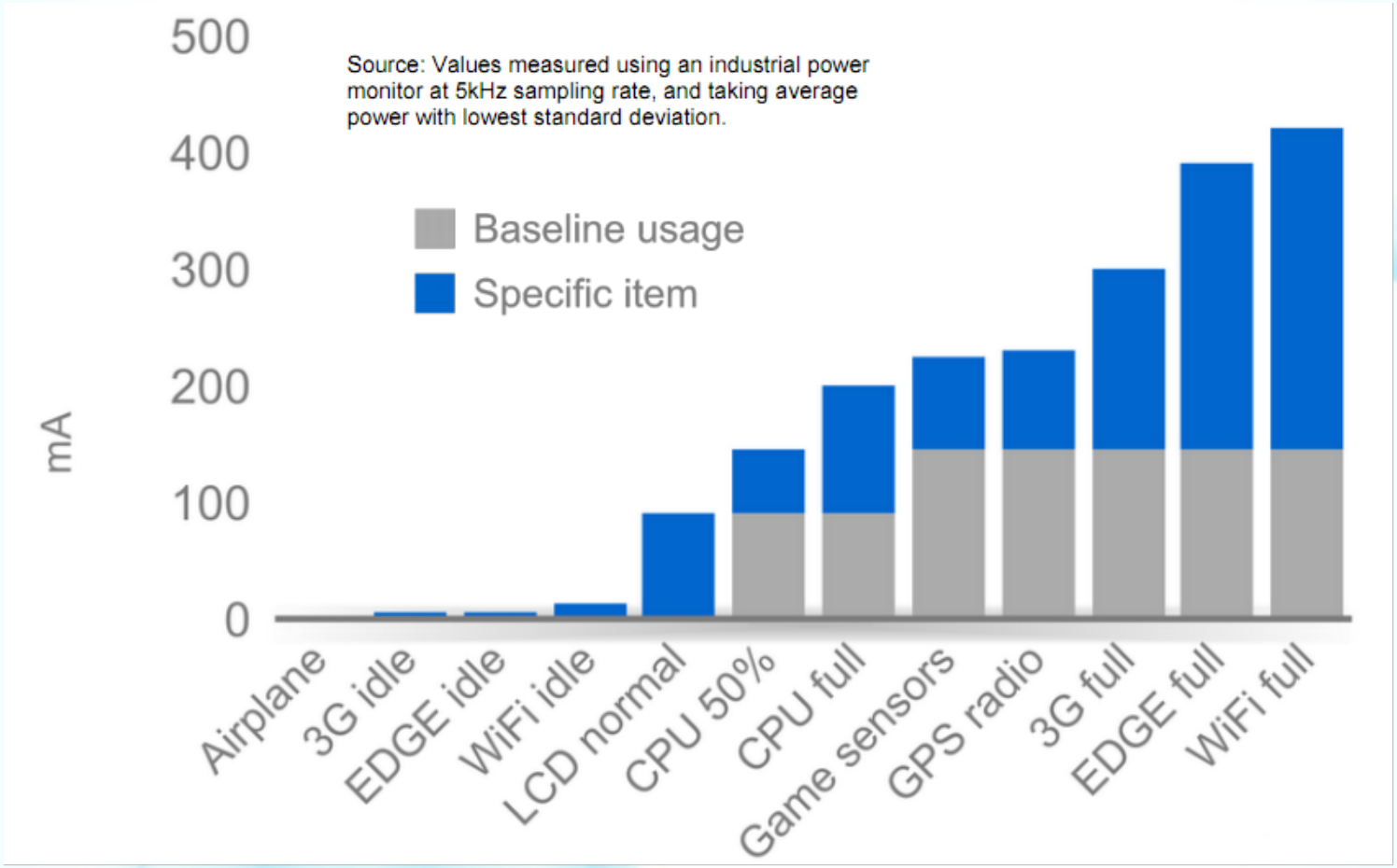
1. 刷微博
2. 看小说
3. 看视频
4. 玩游戏
5. 办公
6. 查地图




由于电池的容量没有大幅度提升,使用Android手机会越来越耗电。电池的可用时间极大的缩短。

有哪些地方在耗电？

# 手机中的电老虎



# 手机中的电老虎

1. 网络、GPS、传感器和CPU等
2. 不合理的使用屏幕唤醒wakelock
3. 后台应用过多，频繁进行数据更新操作
4. 不合理的设置屏幕亮度值和屏幕超时值等
5. 启动多个service
6. 创建大量的临时对象

# 不同网络数据传输消耗

- 传输6MB数据:
  - EDGE (90kbps): 300mA * 9.1 min = 45 mAh
  - 3G (300kbps): 210mA * 2.7 min = 9.5 mAh
  - WiFi (1Mbps): 330mA * 48 sec = 4.4 mAh

# 怎样编写更省电

# 网络

```
 // 只在WiFi或3G下使用网络
NetworkInfo info;
int netType = info.getType();
int netSubtype = info.getSubtype();
if (netType == ConnectivityManager.TYPE_WIFI) {

     return info.isConnected();

} else if (netType == ConnectivityManager.TYPE_MOBILE
             && netSubtype == TelephonyManager.NETWORK_TYPE_UMTS
             && !mTelephony.isNetworkRoaming()) {

     return info.isConnected();

} else {
     return false;
}
```

# 设置网络连接超时时间

正确

```
URL httpUrl = new URL(url);
HttpURLConnection connection =
    (HttpURLConnection)httpUrl.openConnection();
connection.setConnectTimeout(5000) // unit milliseconds
```

错误

```
URL httpUrl = new URL(url);
HttpURLConnection connection =
    (HttpURLConnection)httpUrl.openConnection();
//connection.setConnectTimeout(timeout) //default 0s
```

# 尽量使用GZIP进行文本数据传输

```
import java.util.zip.GZIPInputStream;
HttpGet request = new
        HttpGet("http://example.com/gzipcontent");
HttpResponse resp = new
        DefaultHttpClient().execute(request);
HttpEntity entity = response.getEntity();
InputStream compressed = entity.getContent();
InputStream rawData = new
        GZIPInputStream(compressed);
```

# 缓存文件到本地

```
HttpClient httpclient = new DefaultHttpClient();
httpclient.getParams().setParameter(
                CoreConnectionPNames.SO_TIMEOUT, 5000);
HttpGet httpget = new       HttpGet("http://example.com/
    pictures");
HttpResponse response = httpclient.execute(httpget);
Header[] header = response.getHeaders("ETag");
if(header != null && header.length > 0){
   String catch = header[0].toString();
}
```

# 减少连接次数

重用已经存在的网络连接比新建立一个连接通常更有效率的

由于http协议的“三次握手”

# 判断GPS开关状态

正确

```
String provider = Settings.Secure.getString(

    context.getContentResolver(),Settings.Secure.LOCATION_PROVIDE
    RS_ALLOWED);
if (provider != null && provider.contains("gps")) {
   return true;
}
```

错误

```
LocationManager lManager = (LocationManager)
    context.getSystemService(Context.LOCATION_SERVICE);
lManager .isProviderEnabled(LocationManager.GPS_PROVIDER);
```

# Sensor

正确

```
 @Override
protected void onResume() {

    mSensorManager.registerListener(this,mSensorManager.getDefaultS
    ensor(Sensor.TYPE_ACCELEROMETER),SensorManager.SENSOR
    _DELAY_NORMAL);
}

@Override
protected void onPause() {
    mSensorManager.unregisterListener(this);
}
```

# Sensor

错误

```
public final class BkService extends Service implements
        SensorEventListener{
 @Override
protected void onStart() {

    mSensorManager.registerListener(this,mSensorManager.getDefaultS
    ensor(Sensor.TYPE_ACCELEROMETER),SensorManager.SENSOR
    _DELAY_NORMAL);
}

@Override
protected void onDestroy() {
    mSensorManager.unregisterListener(this);
}
```

# 调整定时更新的频率

```
int alarmType = AlarmManager.ELAPSED_REALTIME;
long interval = AlarmManager.INTERVAL_HOUR;
long start = System.currentTimeMillis() + interval;
alarmManager.setInexactRepeating(alarmType, start,
    interval, pi);
```

如果可以，请设置提醒的类型为ELAPSED_REALTIME or RTC 而不是_WAKEUP。这样在系统睡眠的时候不会唤醒CPU。

# 回收java对象

有些对象的回收不能依赖于JAVA的GC

XmlPullParserFactory and BitmapFactory

Matcher.reset(newString) for regex

StringBuilder.setLength(0)

# 使用软引用

正确

```
ApplicationInfo ai;
PackageManager mPm;
ai = mPm.getApplicationInfo(app.pkgName, 0);
app.icon = new SoftReference<Drawable>(ai.loadIcon(mPm));
holder.iconView.setImageDrawable(app.icon.get());
```

错误

```
ai = mPm.getApplicationInfo(app.pkgName, 0);
Drawable icon= ai.loadIcon(mPm);
holder.iconView.setImageDrawable(icon);
```

# 按需操控广播接收者

```
ComponentName receiver = new ComponentName(context,
    myReceiver.class);
PackageManager pm = context.getPackageManager();
pm.setComponentEnabledSetting(receiver,
    PackageManager.COMPONENT_ENABLED_STATE_EN
    ABLED,  PackageManager.DONT_KILL_APP)
```

# 数据库使用事务

```
SQLiteDatabase db = mHelper.getWritableDatabase();
db.beginTransaction();
try{
    for (int i = 0; i < usageHisList.size(); i++) {
        db.insert(table,null, values);
    }
     db.setTransactionSuccessful();
}catch(Exception ex){
}
db.endTransaction();
```

# 动画

正确

```
@Override
protected void onResume() {
    image.startAnimation(mChargingAnim); //开始动画
}

 @Override
protected void onPause() {
   image.clearAnimation(); //结束动画

}
```

# 动画

错误

```
@Override
protected void onCreate(Bundle b) {
    image.startAnimation(mChargingAnim); //开始动画
}

 @Override
protected void onDestory() {
   image.clearAnimation(); //结束动画

}
```

# 尽量使用 application context

在android中有两种context，一种是 application context，一种是activity context。

```
public class BatteryActivity extends Activity {
    private static Drawable mBackground;
    protected void onCreate(Bundle state) {
        super.onCreate(state);
        TextView label = new TextView(this.getApplicationContext());
        label.setText("It is good !");
        if (mBackground == null) {
            mBackground = getDrawable(R.drawable.battery_info);
        }
    label.setBackgroundDrawable(mBackground);//drawable attached to a view
    setContentView(label);
    }
}
```

# 尽量减少布局层次

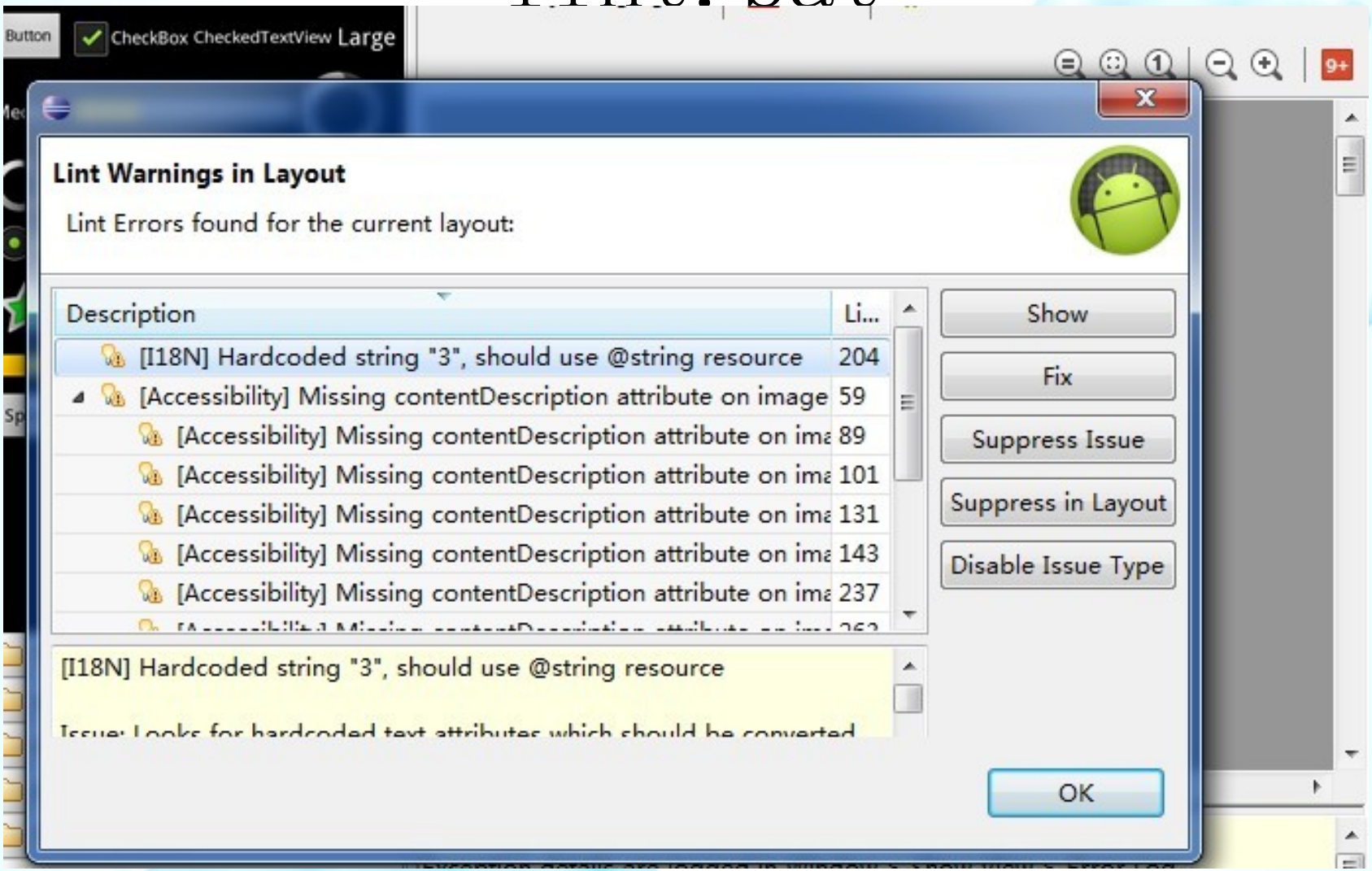
1. Textview +Imageview → Textview +icon（android:drawableXXX）
2. 多个LinearLayout→单个RelativeLayout
3. 通过<include /> 标签来重用layout代码
4. 使用<merge /> 标签来减少视图层级结构

# 性能分析工具介绍

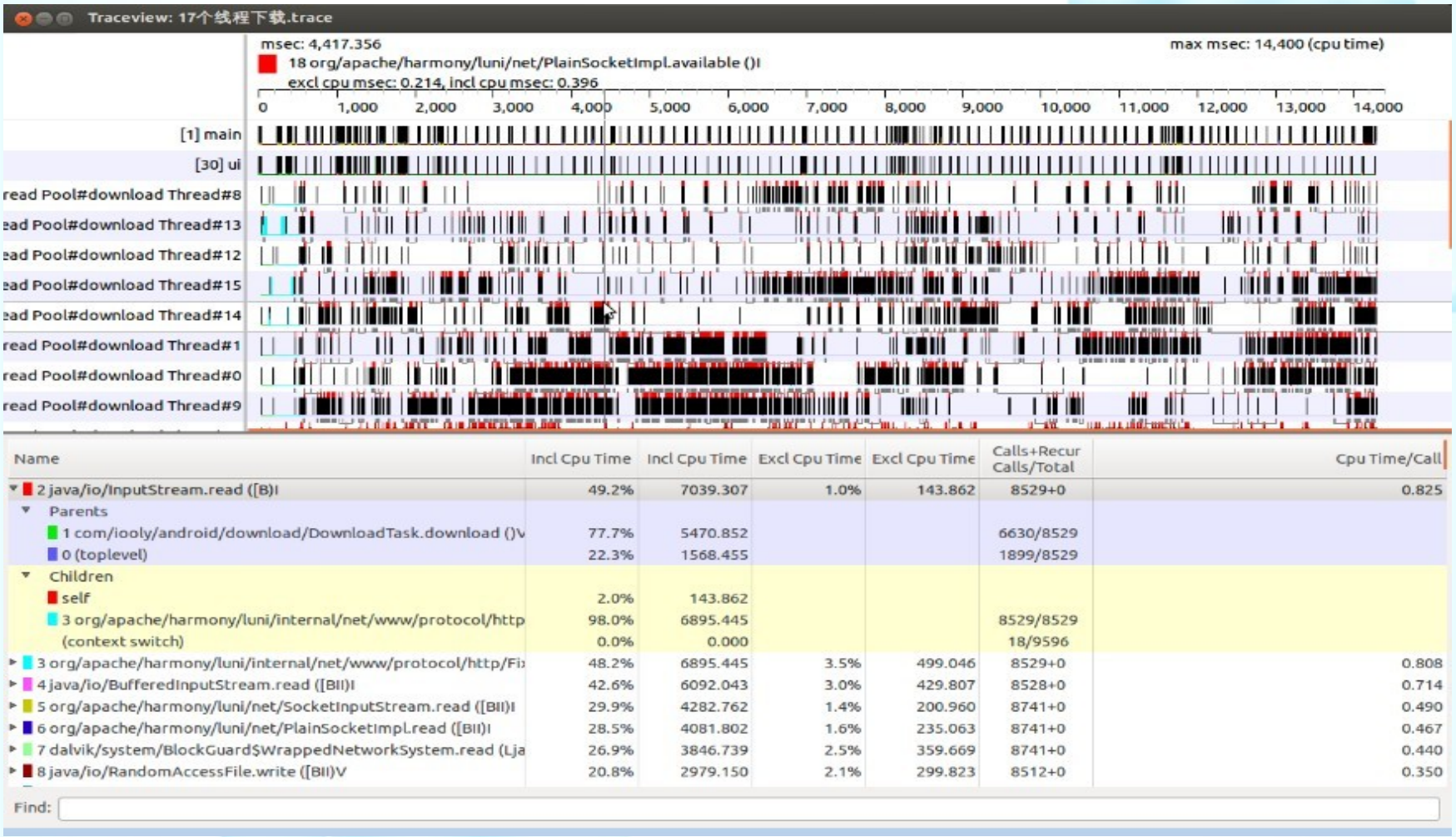
- tools/hierarchyviewer.bat查看UI布局设计结构和各种属性的信息
- tools/lint.bat分析布局文件合理性，给出优化建议
- tools/traceview.bat查看函数以及子函数运行时间
- tools/systrace/systrace.py可以从Linux内核中直接收集数据，来帮助诊断性能问题

# hierarchyviewer.bat

# lint.bat



# traceview.bat




| Name | Incl Cpu Time | Incl Cpu Time | Excl Cpu Time | Excl Cpu Time | Calls+Recur Calls/Total | Cpu Time/Call |
|---|---|---|---|---|---|---|
| 2 java/io/InputStream.read ([B)I | 49.2% | 7039.307 | 1.0% | 143.862 | 8529+0 | 0.825 |
| Parents | | | | | | |
| 1 com/iooly/android/download/DownloadTask.download ()V | 77.7% | 5470.852 | | | 6630/8529 | |
| 0 (toplevel) | 22.3% | 1568.455 | | | 1899/8529 | |
| Children | | | | | | |
| self | 2.0% | 143.862 | | | | |
| 3 org/apache/harmony/luni/internal/net/www/protocol/http | 98.0% | 6895.445 | | | 8529/8529 | |
| (context switch) | 0.0% | 0.000 | | | 18/9596 | |
| 3 org/apache/harmony/luni/internal/net/www/protocol/http/Fi | 48.2% | 6895.445 | 3.5% | 499.046 | 8529+0 | 0.808 |
| 4 java/io/BufferedInputStream.read ([BII)I | 42.6% | 6092.043 | 3.0% | 429.807 | 8528+0 | 0.714 |
| 5 org/apache/harmony/luni/net/SocketInputStream.read ([BII)I | 29.9% | 4282.762 | 1.4% | 200.960 | 8741+0 | 0.490 |
| 6 org/apache/harmony/luni/net/PlainSocketImpl.read ([BII)I | 28.5% | 4081.802 | 1.6% | 235.063 | 8741+0 | 0.467 |
| 7 dalvik/system/BlockGuard$WrappedNetworkSystem.read (Lja | 26.9% | 3846.739 | 2.5% | 359.669 | 8741+0 | 0.440 |
| 8 java/io/RandomAccessFile.write ([BII)V | 20.8% | 2979.150 | 2.1% | 299.823 | 8512+0 | 0.350 |

Find:

# 总结

- 为什么要省电？
- 有哪些地方在耗电？
- 怎样编写更省电？
- 性能分析工具介绍

# 感谢大家关注

# Q&A

联系方式：

官方微博@ 金山电池医生

微信号：dianchixiaoyisheng